ISSN (Online): 2348-8557

International Journal of Computer Informatics & Technological Engineering

VOLUME -1, ISSUE -1, MARCH- APRIL, 2014

PAPER ID: 2014/M-A/IJCITE/V1-E1-24

IJCITE

ANDROID THREADING: A NEW PARADIGM, CONCEPTS, USAGE

Dr. Mijal Mistry

Assistant Professor, ISTAR

V.V.Nagar, Gujarat

Abstract: **The paper shows the problem which were faced by the android developer after publishing the application. Some application which involves database manipulation might not work on all devices. Some devices directly not allowed network operations which involves cross platform communication also. Some applications work on smoothly on some devices and not in other devices. Developer needs to find the solution for this type of problem. This paper shows way to tackle the issue and proposed a solution using which application will run perfectly and efficiently on all the devices.**

Keywords: **Android, Threading, Multi-Threading, Handlers**

I. INTRODUCTION

Threads are the important part of any multitasking operating system and can be consider as another small-processes running within a main process. The main idea is to provide a way for analogous executions for any application that is using threads. [1] Android application works on different way. Once it starts first time, the operating system a thread is created in which all the required components are loaded. When the request for the components is made then it accesses the components from the thread. Such type of thread is considered as Main thread. The main activity which is performed by main thread is related to the user interface. User interface can use Event handling, communication with other components and activities. [1] Sometimes it may happen that without using threading, if heavy burden is given to main thread of the application, then it may delay the processing and may crash the application. Sometimes it also blocks the entire application. It will also give a message like "Application is not responding" with Wait and OK option. This type of working application is normally not acceptable in real life. This can be avoided if threading is used. Developer can use multiple threads to perform various activities independently. Threads can also communicate with each other to carry out the task. [1]

If the activity involves intensive database operations then the speed of the application often slows down. To improve the application performance, threading can be useful. Overall application can be divided into minor modules which run simultaneously and carry out the operations. CPU with multiple processors functionality can run threads concurrently. [2] Application developed in android works little bit different. It gathers various events in a row and processed using Looper class. Below figure 1 shows the basic structure of looper class. [3]

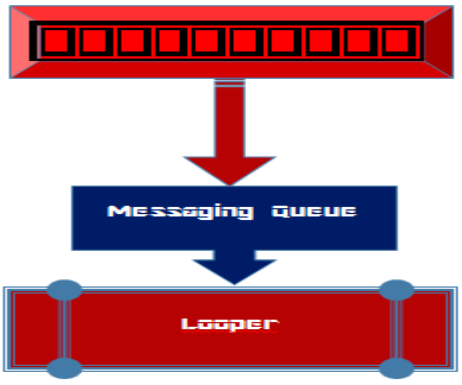

Figure 1 Structure of looper class

ISSN (Online): 2348-8557

International Journal of Computer Informatics & Technological Engineering

VOLUME -1, ISSUE -1, MARCH- APRIL, 2014

PAPER ID: 2014/M-A/IJCITE/V1-E1-24

IJCITE

Developer write code without synchronization then the code execute one by one. It does not achieve concurrent execution. [3] If android application is accessing the data from the live web services then the application remains halt until the task is successfully complete. [3]

For fascinating and fast performance activities within the application run concurrently. Network operations, data manipulation with various databases, some mathematical complex scenarios are the points where application is likely slow down. [3]

II. THREADING

Android provides facility for threading which helps to overcome the above mentioned scenarios. In application developer can use Thread class to carry out concurrent executions. Android can use java.util.concurrent package to accomplish task in background. One can use the android.OS.Handler class or the AsyncTasks classes. There are also other various ways for using it like use of Loader class, Fragments, web services, normal services and so on. [3]

Android OS is designed with the help of Linux, so it uses few mechanism same like Linux. For e.g. Thread management, running the application and so on. When developer creates basic or advanced application using android, by default it uses single thread which is normally considered as main thread. [4] When time consuming task is being performed on activity, sometimes user may notice an unpleasant user experience. Sometimes used stops using the application as a result. Sometimes application may hanged. There are several operations can be running at the same time for the app which may need to run separately.

III. IMPLEMENTED SYSTEM

In practical scenario, I have created one application which access the user credentials and verifies the user's authenticity. When I install it on Samsung device it works fine without any issues, but when I install it on some other devices like LG, Lenovo then it is not working correctly. I started debugging the issue from the scratch and check each and every step to find out where is the problem. The issue is related to the network operations. I also checked the permissions for android operations. I have given the necessary permission to application. The main reason for the issue is, the code for database operation is written on the Main Activity of android. After doing some research found the solution for the problem. The problem with the application is that, it is not using thread. Before I walk through to the remaining process for using Thread, below list shows the process for any application which should be carried out as a separate process. They are:

- Networking –It tries to access some network related services or performing some network operation.
- Database operations – It tries to perform database calculations which require connecting the database, fires the query and showing the result.
- Heavy calculations – some heavy mathematical calculations can be carried out.
- Object's long initialization – sometimes android app is in process for initializing long object values or it loads the long data.

For most of the people, thread is the confusing topic. Here, more efficient and easy way is shown to write the code for Thread.

There is couple of options to use the thread as a new thread. One can divide Thread and override run**()** method. Other option is create a new Thread and add a **Runnable** to the constructor. In both the cases, **start()** method is required to execute for new thread. Each Thread has its own priority defined. The priority is measured in integer form. Based on the priority it is available in OS. A child thread can b default inherits the parent's priority. Once can set the priority of thread using set **Priority (int)** method. The parameter is of an integer type which is helpful for OS to set the priority.

Firstly, we need to create the object of Thread class. This class is available in **java.lang** namespace.

Let us examine the code step by step. Below line creates the object of Thread class with override the **Run()**method.

```
Thread background = new Thread(new Runnable(){
};
```

ISSN (Online): 2348-8557

International Journal of Computer Informatics & Technological Engineering

VOLUME -1, ISSUE -1, MARCH- APRIL, 2014

PAPER ID: 2014/M-A/IJCITE/V1-E1-24

IJCITE

Background is the name of the object that is running in the background. Next we are creating the object of**HttpClient** using below line.

```
private final HttpClient Client = new DefaultHttpCli
ent();
```

Our next task is creating the **Run()** method which is running in the background. In this method we are taking the objects of **Textviews** of the forms and assigning it to the local variable (username and password) which contains the entered text of user. Using the values a URL will be formed which will be used in **HttpGet** for querying the server. Below line of code shows it.

```
HttpGet httpget = new HttpGet("http://1922.168.1.1/Android/Check/master/master");
```

Next task is accepting the response return from the server. For performing this operation a object of**ResponseHandler<T>** interface is created. Below code snippet shows it.

```
ResponseHandler<String> responseHandler = new BasicResponseHandler();
```

In place of **<T>,** one can add any type of object for conversion. Next is executing the client request with necessary values and parameters. Below code do it. The return result is stored into the **SetServerString**variable. It is of string type.

```
SetServerString = Client.execute(httpget, responseHandler);
```

Handler objects are used for performing inter communication between various threads.

If application has two activities and they want to navigate with each other with some values then Handler comes into picture. It must request a message token using the obtainMessage() method. The handleMessage() method is used to handle messages that arriving to the main thread.

There are two main usages of Handler: [7]

- It is used to organize various messages to be executed.
- It is used to insert the task which is going to be carried out on other thread.

Once we have server response we will pass this message to the **ThreadMessage** function. This function will convert the return string into the Android Message using **Android.os** class. The message is passed to the bundle class. The main functionality of the bundle class is mapping from String values to various **Parcelable** types. The whole message object is passed to the method which actually executes in background. Below piece of code shows it.

```
private void threadMsg(String msg) {
        if (!msg.equals(null) &&
!msg.equals("")) {
            Message msgObj
= handler.obtainMessage();
            Bundle b = new Bundle();
            b.putString("message", msg);
            msgObj.setData(b);
            handler.sendMessage(msgObj);
        }
    }
```

Next and final task is calling the **SendMessage** method. This method actually contains the code for performing activities like, checking whether the values which we received from the server is correct or not, writing the code for performing the activities like (moving to next activity, giving the information message for incorrect username or password etc.). Below line of code shows it.

```
private final Handler handler = new Handler() {

public void handleMessage(Message msg) {

 String aResponse =
msg.getData().getString("message");

  if ((null != aResponse)) {

    if(!aResponse.contains("success")){

Toast.makeText(getApplicationContext(),"Credential
s are matched", Toast.LENGTH_LONG).show();

            }
    else
            {
Toast.makeText(getApplicationContext(),"Credential
s are not matched", Toast.LENGTH_LONG).show();
```

ISSN (Online): 2348-8557

International Journal of Computer Informatics & Technological Engineering
VOLUME -1, ISSUE -1, MARCH- APRIL, 2014
PAPER ID: 2014/M-A/IJCITE/V1-E1-24

IJCITE

```
            }
        }
    }
};
```

Once the code is over our primary and main task is starting the thread. Thread can be started using the below line of code.

background.start();

IV. ADVANTAGES OF THREAD

- It helps to improve overall performance of the application
- It also minimized the system resource usage by using thread
- Application can use multiple applications simultaneously
- It also simplifies the program structure
- Simplified coding of remote procedure calls and conversations
- Threading provides an useful abstraction of concurrent execution.

V. CONCLUSION

This paper shows the way using which one can solve the issue of network operations problem. It also shows the way to write the code in such a way that it won't create more processing burden.

VI. REFERENCES

[1]http://www.techotopia.com/index.php/A_Basic_Overview_of_Android_Threads_and_Thread_handlers

[2]http://developer.android.com/training/multiple-threads/index.html

[3]http://www.vogella.com/tutorials/AndroidBackgroundProcessing/article.html

[4]http://developer.samsung.com/android/technical-docs/Basics-of-multi-threading-in-Android

[5]http://csharpdotnet.wordpress.com/2008/11/03/advantages-of-multithreading/

[6]http://careerride.com/NET-Th-advantages-disadvantages-of-multithreading.aspx

[7]http://r4r.co.in/java/android/basic/tutorial/Android/MultiThread_Andriod.shtml